Curso : 2018/19

30218 - Tecnología de programación

Información del Plan Docente

Año académico:	2018/19
Asignatura:	30218 - Tecnología de programación
Centro académico:	110 - Escuela de Ingeniería y Arquitectura 326 - Escuela Universitaria Politécnica de Teruel
Titulación:	439 - Graduado en Ingeniería Informática 443 - Graduado en Ingeniería Informática
Créditos:	6.0
Curso:	443 - Graduado en Ingeniería Informática: 2 439 - Graduado en Ingeniería Informática: 2
Periodo de impartición:	Segundo Semestre
Clase de asignatura:	Obligatoria
Módulo:	---

Información Básica

Objetivos de la asignatura

La asignatura y sus resultados previstos responden a los siguientes planteamientos y objetivos:

Tras haber adquirido los conocimientos básicos de programación, esta asignatura busca consolidar este conocimiento a través de conceptos más avanzados que aparecen en distintos paradigmas y lenguajes de programación.

La asignatura tiene un marcado carácter aplicado. El alumno aprenderá los conceptos avanzados de programación orientada a objetos a partir de un conjunto de problemas presentes en el desarrollo actual de software.

Al mismo tiempo, el alumno conocerá y aplicará otros paradigmas y lenguajes, en especial el paradigma de la Programación Funcional.

Contexto y sentido de la asignatura en la titulación

Tecnología de Programación aparece después de Programación I, Programación II y Estructuras de Datos y Algoritmos cuando el alumno está preparado para abordar tecnologías y conceptos de programación avanzados.

Es una asignatura obligatoria englobada en la materia de formación común en Programación y Computación.

Recomendaciones para cursar la asignatura

El alumno que curse esta asignatura ha de contar con una formación básica en programación.

Competencias y resultados de aprendizaje

Competencias

Al superar la asignatura, el estudiante será más competente para...

Conocer y aplicar los procedimientos algorítmicos básicos de las tecnologías informáticas para diseñar soluciones a problemas, analizando la idoneidad y complejidad de los algoritmos propuestos.

Analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados.

Concebir, diseñar y desarrollar proyectos de Ingeniería.

Resolver problemas y tomar decisiones con iniciativa, creatividad y razonamiento crítico.

Usar las técnicas, habilidades y herramientas de la Ingeniería necesarias para la práctica de la misma.

Aprender de forma continuada y desarrollar estrategias de aprendizaje autónomo.

Aplicar las tecnologías de la información y las comunicaciones en la Ingeniería.

Resultados de aprendizaje

El estudiante, para superar esta asignatura, deberá demostrar los siguientes resultados:

R12. Capacidad para desarrollar programas complejos en leguajes orientados a objetos, y utilizar la programación orientada a objetos en desarrollos que incorporen interfaces gráficos de usuario, sistemas de gestión de eventos o accesos a bases de datos y a recursos distribuidos en la red.

R13. Conocer y comprender la sintaxis y la semántica de un lenguaje de programación funcional.

R14. Desarrollar programas escritos en un lenguaje funcional.

R15. Tener una perspectiva de otros paradigmas y lenguajes de programación.

La asignatura pretende avanzar un paso más en las técnicas de programación imperativa aprendidas en los tres primeros cuatrimestres del Grado:

- La formalización de las bases de la POO: clases
- La utilización de técnicas más avanzadas en POO: herencia, polimorfismo, programación genérica.
- La aplicación de esas técnicas al desarrollo de software: diseño de clases y jerarquías de herencia

Y además introducir con detalle un nuevo paradigma de programación: la Programación Funcional, trabajando las ideas de

- Recursividad
- Funciones de orden superior

Y por último, se da un ligero repaso a otros paradigmas y tecnologías de programación:

- Paradigma Lógico y Lenguajes dinámicos.

Importancia de los resultados de aprendizaje

Aprender a programar es esencial para un ingeniero informático. Lo que aprenda en esta asignatura, que complementa a lo ya aprendido en asignaturas anteriores del bloque de programación, dotará al alumno de una perspectiva global de las tecnologías de programación y su aplicación a distintos contextos. Los conceptos que se ven en la asignatura le permitirán tener un mayor abanico en una herramienta fundamental como el lenguaje o la tecnología de programación elegidos para resolver un problema.

Evaluación

Tipo de pruebas y su valor sobre la nota final y criterios de evaluación para cada prueba

El estudiante deberá demostrar que ha alcanzado los resultados de aprendizaje previstos mediante las siguientes actividades de evaluación

A lo largo del semestre se desarrollarán clases de prácticas en laboratorio, para las que se formarán equipos integrados como máximo por dos alumnos. Con los trabajos prácticos de programación se realizará un seguimiento del trabajo realizado por los alumnos durante el semestre y del progreso de su aprendizaje. Los trabajos presentados por el alumno se calificarán con una nota cuantitativa de 0 a 10. Para obtener dichas notas se valorará el funcionamiento de los programas según especificaciones, la calidad de su diseño y su presentación, la adecuada aplicación de los métodos de resolución, el tiempo empleado, así como la capacidad de los integrantes del equipo para explicar y justificar el diseño realizado. Los alumnos que hayan cumplido con los plazos de entrega fijados para los trabajos prácticos de programación, y hayan demostrado en ellos un nivel de aprovechamiento y calidad de resultados adecuados, obteniendo en la valoración de su trabajo práctico al menos la nota mínima especificada, serán exentos de la realización del examen práctico de programación en laboratorio.

Examen práctico e individual de programación, en laboratorio. En el examen práctico se le plantearán al alumno ejercicios de programación de naturaleza similar a los realizados en las prácticas o vistos en clase. Se calificará con una nota de 0 a 10, para la que se valorará el correcto funcionamiento y rendimiento de los programas según especificaciones, la calidad de su diseño, la adecuada aplicación de los métodos de resolución y el tiempo empleado. Los alumnos que resulten exentos de la realización de este examen y opten por presentarse al mismo, renunciarán con ello a la nota obtenida con la entrega de sus trabajos prácticos, de forma irreversible.

Examen escrito en el que se deberán resolver problemas de programación y, en su caso, responder preguntas conceptuales o resolver algún ejercicio. Se calificará con una nota de 0 a 10. En general, se valorará la calidad y claridad de las respuestas y soluciones propuestas, su adecuación a las especificaciones y restricciones planteadas, la calidad del diseño, la adecuada aplicación de los métodos de resolución y el tiempo empleado.

Ponderación de las actividades de evaluación

Las calificaciones obtenidas en las dos partes en la primera convocatoria se guardan para la segunda convocatoria en el caso de que el alumno no logre aprobar la asignatura.

A continuación se describe el método de ponderación de las distintas pruebas en los Campus Río Ebro y Campus de Teruel:

En la Escuela de Ingeniería y Arquitectura del Campus Rio Ebro:

Las dos partes de la prueba se ponderarán de la siguiente forma:

- Examen escrito de teoría y problemas: 60%.
- Examen práctico de programación: 40%.

Es necesario una calificación mínima de 5.0 puntos en el examen escrito para aprobar la asignatura. Si la calificación en el examen escrito es inferior a 5.0, la calificación del alumno en la asignatura es la obtenida en dicho examen. Si, por el contrario, esa calificación es igual o superior a 5.0 la calificación del alumno en la asignatura se obtiene como suma ponderada de las calificaciones del examen escrito (con ponderación del 60%) y del examen práctico (con ponderación del 40%).

En la Escuela Universitaria Politécnica del Campus de Teruel:

Las dos partes de la prueba se ponderarán de la siguiente forma:

- Examen escrito de teoría y problemas: 50%.
- Examen práctico de programación: 50%.

Es necesario una calificación mínima de 4.0 puntos en el examen escrito para aprobar la asignatura. Si la calificación en el examen escrito es inferior a 4.0, la calificación del alumno en la asignatura es la obtenida en dicho examen. Si, por el contrario, esa calificación es igual o superior a 4.0 la calificación del alumno en la asignatura se obtiene como suma ponderada de las calificaciones del examen escrito (con ponderación del 50%) y del examen práctico (con ponderación del 50%).

Metodología, actividades de aprendizaje, programa y recursos

Presentación metodológica general

El proceso de aprendizaje que se ha diseñado para esta asignatura se basa en lo siguiente:

En las clases impartidas en el aula se desarrollará el temario de la asignatura.

En las clases de problemas se resolverán problemas de aplicación de los conceptos y técnicas presentadas en el programa de la asignatura.

Las sesiones de prácticas de desarrollan en un laboratorio informático. En cada sesión el alumno deberá realizar un trabajo de programación directamente relacionado con los temas estudiados en la asignatura.

Actividades de aprendizaje

El programa que se ofrece al estudiante para ayudarle a lograr los resultados previstos comprende las siguientes actividades...

1. El estudio y trabajo continuado desde el primer día de clase.
2. El aprendizaje de conceptos y metodologías para el análisis y el diseño de programas correctos y eficientes a través de las clases magistrales, en las que se favorecerá la participación de los alumnos.
3. La aplicación de tales conocimientos al diseño y análisis de algoritmos y programas en las clases de problemas. En estas clases los alumnos desempeñarán un papel activo en la discusión y resolución de los problemas.
4. Las clases de prácticas en laboratorio en las que el alumno aprenderá la tecnología necesaria para desarrollar pequeños proyectos de programación utilizando un lenguaje de programación determinado.
5. El trabajo en equipo desarrollando algún pequeño proyecto de programación cuyo resultado se plasma en la entrega de programas resultantes convenientemente diseñados y documentados.
6. Un trabajo continuado en el que se conjugue la comprensión de conceptos, el análisis y la resolución de problemas de programación utilizando "lápiz y papel" y la puesta a punto en computador de algunos pequeños proyectos de programación.

Programa

l programa de la asignatura se divide en dos grandes bloques, uno dedicado a la programación orientada a objetos y otro a la programación funcional. Existe un tercer bloque muy corto destinado a la presentación a nivel de charlas de otros paradigmas de programación.

- **Programación Orientada a Objetos**
  - Clases
  - Herencia y Polimorfismo
  - Programación Genérica
  - Contenedores y Estructuras de Datos
  - Inferencia de Tipos - Excepciones
  - Diseño Orientado a Objetos- Patrones de Diseño
  - Programación orientada a eventos aplicada al diseño de interfaces gráficos de usuario

- Acceso a bases de datos y recursos distribuidos en red

- **Programación Funcional**
  - Introducción a la Programación Funcional
  - Lenguajes para Programación Funcional
  - Datos y Tipos
  - Expresiones y Funciones
  - Recursividad
  - Listas y Funciones de Orden Superior
- **Otros Paradigmas de Programación**
  - Paradigma Lógico y Lenguajes dinámicos

Planificación de las actividades de aprendizaje y calendario de fechas clave

Calendario de sesiones presenciales y presentación de trabajos

La organización docente de la asignatura prevista Escuela de Ingeniería y Arquitectura del Campus Rio Ebro es la siguiente:

- Clases teóricas: 2 horas semanales
- Clases de problemas: 1 hora semanal
- Clases prácticas de laboratorio: siete sesiones de 2 horas, una sesión cada dos semanas. Son sesiones de trabajo de programación en laboratorio, tuteladas por un profesor, en las que participan los alumnos de cada uno de los subgrupos en los que se divide el grupo.

La organización docente de la asignatura prevista en Escuela Universitaria Politécnica del Campus de Teruel es la siguiente:

- Clases teóricas: 2 horas semanales
- Clases de problemas y prácticas de laboratorio: 2 horas semanales, Son sesiones de trabajo de programación en laboratorio, tuteladas por un profesor, en las que participan los alumnos de cada uno de los subgrupos en los que se divide el grupo.

Presentación de trabajos objeto de evaluación:

- Los problemas y ejercicios que se propongan para ser resueltos individualmente en las clases de problemas se entregarán en las mismas clases de problemas en los que se planteen.
- El proyecto de programación en equipo será entregado en la fecha que sea anunciada al proponer los trabajos.

Trabajo del estudiante

La dedicación del estudiante para alcanzar los resultados de aprendizaje en esta asignatura se estima en 156 horas aproximadamente, distribuidas del siguiente modo:

- 60 horas de actividades presenciales (clases teóricas, de problemas y prácticas en laboratorio)
- 30 horas de trabajo de programación en equipo
- 60 horas de estudio personal efectivo (estudio de apuntes y textos, resolución de problemas, preparación clases y prácticas, desarrollo de programas)
- 6 horas de examen final de teoría escrito y de prácticas en laboratorio

El calendario de exámenes y las fechas de entrega de trabajos se anunciará con suficiente antelación.

Bibliografía y recursos recomendados

Zaragoza:

BB	1. Bird, Richard. Introducción a la programación funcional c Haskell / Richard Bird . - 1a ed. en español Madrid [etc.] : Prentice Hall, D.L. 2000
BB	2. Eckel, Bruce. Piensa en Java / Bruce Eckel ; traducción, González Barturen ; revisión técnica, Javier Parra Fuente, Ricardo Lozano Quesada ; coordinación general y revisión técnica, Luis Joyanes Aguilar . - 2ª ed. Madrid [etc.] : Prent Hall, D.L. 2002
BB	3. Design patterns : Elements of reusable object-oriented software / Erich Gamma...[et al.] . - 19th pr. Reading, Massachusetts : Addison-Wesley, 2000
BC	Bloch, Joshua. Effective Java / Joshua Bloch. 2nd ed. Addison-Wesley, 2008
BC	The JFC swing tutorial : A guide to constructing GUIs / Katl Walrath [et al.] . - 2nd ed. Boston [etc.] : Addison-Wesley, c 2004